Amendments to the claims (this listing replaces all prior versions):

1. (currently amended) A method comprising:

identifying a plurality of memory resources for pushing data to and pulling data from a processing agent;

~~during a read phase;~~

when pushing data to the processing agent,

using a push bus arbiter to arbitrate use of a push bus by the memory resources[[;]] , and

pushing the data from the memory resources to the processing agent through the push bus, the memory resources obtaining access to the push bus based on arbitration by the push bus arbiter; and

~~during a write phase;~~

when pulling data from the processing agent,

using a pull bus arbiter to arbitrate use of a pull bus by the memory resources[[;]],

executing a context [[;]] ,

issuing a write command [[;]] ,

loading data into an output transfer ~~register~~ memory of the ~~programming~~ processing agent[[;]] ,

setting the output transfer ~~register~~ memory to a read-only state [[;]] , and

pulling the data from the output transfer ~~register~~ memory and transferring to the memory resources through the pull bus, the memory resources obtaining access to the pull bus based on arbitration by the pull bus arbiter.

2. (currently amended) The method of claim 1 further comprising:

establishing a plurality of contexts on the ~~programming~~ processing agent and maintaining program counters and context relative registers.

3. (currently amended) The method of claim 2 wherein the ~~programming~~ processing agent executes a context and issues a read command to a memory controller in a read phase.

4. (previously presented) The method of claim 3 wherein the memory controller processes the read command to be sent to one of the memory resources.

5. (original) The method of claim 4 wherein the context is swapped out if the read data is required to continue the execution of the context.

6. (currently amended) The method of claim 5 wherein after the memory controller has completed the processing of the read command, the memory controller pushes the data to an input transfer ~~register~~ memory of the ~~programming~~ processing agent.

7. (currently amended) The method of claim [[6]] 58 wherein after the data has been pushed, the ~~programming~~ processing agent reads the data in the input transfer register and the ~~programming~~ processing agent continues the execution of the context.

8-12. (cancelled)

13. (currently amended) A system comprising:

a plurality of memory resources, each memory resource being associated with a memory controller;

a processing agent to access the memory resources;

a push bus to push data from the memory resources to the processing agent;

a push bus arbiter to arbitrate use of the push bus by the memory resources, the memory resources obtaining access to the push bus based on arbitration by the push bus arbiter;

a pull bus to receive data from the processing agent and to transfer the data to the memory resources; and

a pull bus arbiter to arbitrate use of the pull bus by the memory resources, the memory resources obtaining access to the pull bus based on arbitration by the pull bus arbiter;

wherein ~~one of the memory resources transfers data to the processing agent~~ data are transferred unidirectionally ~~during a read phase~~ on the push bus.

14. (cancelled)

15. (cancelled)

16. (original) The system of claim 13 further comprising a plurality of program counters and a plurality of context relative registers.

17. (original) The system of claim 16 in which the context relative registers are selected from a group comprising of general purpose registers, inter-programming agent registers, static random access memory (SRAM) input transfer registers, dynamic random access memory (DRAM) input transfer registers, SRAM output transfer registers, DRAM output transfer registers, and local memory registers.

18. (currently amended) The system of claim 17 in which the ~~programming~~ processing agent is to execute a context and issue a read command to a memory controller.

19. (original) The system of claim 18 in which the memory controller is to process the read command to be sent to the memory resource.

20. (currently amended) The system of claim 19 in which the ~~programming~~ processing agent is to swap the context out if the read command is required to continue to execution of the context.

21. (currently amended) The method system of claim 20 in which after the read command is processed, the memory controller is to push the data to an input transfer register of the ~~programming~~ processing agent and the ~~programming~~ processing agent is to read the data in the input transfer register and to continue the execution of the context.

22-25. (cancelled)

26. (currently amended) A machine-accessible medium, which when accessed results in a machine performing operations comprising:

identifying a plurality of memory resources for pushing data to and pulling data from a processing agent, each memory resource being associated with a memory controller;

~~during a read phase,~~

when pushing data to the processing agent,

using a push bus arbiter to arbitrate use of a push bus by the memory resources[[;]] ,

pushing the data from the memory resources to the processing agent through the push bus, the memory resources obtaining access to the push bus based on arbitration by the push bus arbiter [[;]] ,

~~during a write phase,~~

when pulling data from the processing agent,

using a pull bus arbiter to arbitrate use of a pull bus by the memory resources[[;]],

executing a context [[;]] ,

issuing a write command [[;]] ,

loading data into an output transfer ~~register~~ memory of the ~~programming~~ processing agent[[;]] ,

setting the output transfer ~~register~~ memory to a read-only state [[;]] , and

pulling data from the output transfer ~~register~~ memory and transferring the data to the memory resources through the pull bus, the memory resources obtaining access to the pull bus based on arbitration by the pull bus arbiter.

27. (currently amended) The machine-accessible medium of claim 26, which when accessed further results in the machine performing operations comprising establishing a plurality of contexts on the ~~programming~~ processing agent and maintaining program counters and context relative registers.

28. (currently amended) The computer program product of claim 26 wherein the ~~programming~~ processing agent in a read phase executes a context and issues a read command to a memory controller.

29. (original) The computer program product of claim 26 wherein the memory controller processes the read command to be sent to the memory resource and the context is swapped out if the read command is required to continue the execution of the context.

30. (cancelled)

31. (cancelled)

32. (cancelled)

33. (previously presented) The method of claim 1 wherein the context is swapped out if the write command is required to continue the execution of the context.

34. (currently amended) The method of claim 33 wherein the memory controller pulls the data from the output transfer ~~register~~ memory and the memory controller sends a signal to the ~~programming~~ processing agent to unlock the output transfer ~~register~~ memory.

35. (currently amended) The method of claim 34 wherein if the context has been swapped out after the output transfer ~~register~~ memory has been unlocked, the context is swapped back in and the ~~programming~~ processing agent continues the execution of the context.

36. (cancelled)

37. (currently amended) A system comprising:

a plurality of memory resources, each memory resource being associated with a memory controller;

a processing agent to access the memory resources;

a push bus to push data from the memory resources to the processing agent;

a push bus arbiter to arbitrate use of the push bus by the memory resources, the memory resources obtaining access to the push bus based on arbitration by the push bus arbiter;

a pull bus to receive data from the processing agent and to transfer the data to the memory resources; and

a pull bus arbiter to arbitrate use of the pull bus by the memory resources, the memory resources obtaining access to the pull bus based on arbitration by the pull bus arbiter;

wherein the processing agent ~~transfers data to one of the memory resources unidirectionally during a write phase~~ is to execute a context and load the data into an output transfer memory of the processing agent, and to issue a write command to a memory controller and in which the output transfer memory is set to a read-only state.

38-39. (cancelled)

40. (currently amended) The system of claim [[39]] 37 in which the ~~programming~~ processing agent is to swap the context out [[if]] of the write command is required to continue to execution of the context.

41. (currently amended) The system of claim 40 in which the memory controller is to push the data from the output transfer ~~register~~ memory and to send a signal to the ~~programming~~ processing agent to unlock the output transfer ~~register~~ memory.

42. (cancelled)

43. (previously presented) The method of claim 1, wherein the memory resources comprise memory controller channels.

44. (previously presented) The system of claim 13, wherein the memory resources comprise memory controller channels.

45. (previously presented) The machine accessible medium of claim 26, wherein the memory resources comprise memory controller channels.

46. (cancelled)

47. (cancelled)

48. (cancelled)

49. (currently amended) A method comprising:

identifying a plurality of memory resources for pushing data to and pulling data from a processing agent;

using a push bus arbiter to arbitrate use of a push bus by the memory resources, data being transferred unidirectionally on the push bus;

pushing the data from the memory resources to the processing agent through the push bus ~~unidirectionally~~, the memory resources obtaining access to the push bus based on arbitration by the push bus arbiter;

using a pull bus arbiter to arbitrate use of a pull bus by the memory resources; and

pulling the data from the processing agent and transferring to the memory resources through the pull bus, the memory resources obtaining access to the pull bus based on arbitration by the pull bus arbiter.

50. (previously presented) The method of claim 49, further comprising executing a context and issuing a read command to a memory controller to read data from one of the memory resources.

51. (previously presented) The method of claim 50, further comprising swapping out the context if the data to be read is required to continue the execution of the context.

52. (currently amended) A method comprising:

identifying a plurality of memory resources for pushing data to and pulling data from a processing agent;

using a push bus arbiter to arbitrate use of a push bus by the memory resources;

pushing the data from the memory resources to the processing agent through the push bus, the memory resources obtaining access to the push bus based on arbitration by the push bus arbiter;

using a pull bus arbiter to arbitrate use of a pull bus by the memory resources, data being transferred on the pull bus unidirectionally; and

pulling the data from the processing agent and transferring to the memory resources through the pull bus ~~unidirectionally~~, the memory resources obtaining access to the pull bus based on arbitration by the pull bus arbiter.

53. (currently amended) The method of claim 52, further comprising ~~operating the~~ executing a context and issuing a write command to a memory controller to write data to one of the memory resources.

54. (previously presented) The method of claim 53, further comprising swapping out the context if completion of the write command is required to continue the execution of the context.

55. (new) A data processor comprising:

a push bus arbiter to arbitrate use of a unidirectional push bus by a plurality of memory resources, the unidirectional push bus allowing the memory resources to push data from the memory resources; and

a pull bus arbiter to arbitrate use of a unidirectional pull bus by the memory resources, the unidirectional pull bus allowing the memory resources to pull data to the memory resources.

56. (new) A data processor comprising:

a plurality of programming engines;

a push bus arbiter to arbitrate use of a push bus by a plurality of external memory resources that are external to the data processor, the push bus arbiter being internal to the data processor, the push bus to push data from the memory resources to an input transfer memory associated with the programming engines; and

a pull bus arbiter to arbitrate use of a pull bus by the external memory resources, the pull bus arbiter being internal to the data processor, the pull bus to pull data from an output transfer memory associated with the programming engines and to transfer the data to the memory resources.

57. (new) The method of claim 1 wherein the output transfer memory comprises an output transfer register.

58. (new) The method of claim 6 wherein the input transfer memory comprises an input transfer register.

59. (new) The machine-accessible medium of claim 26 wherein the output transfer memory comprises an output transfer register.

60. (new) The system of claim 37 wherein the output transfer memory comprises an output transfer register.